


```
0001 0 MODULE SSIK (IDENT = 'V04-000') =
0002 1 BEGIN
0003 1
0004 1
0005 1 *****
0006 1 *
0007 1 *   COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0008 1 *   DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0009 1 *   ALL RIGHTS RESERVED.
0010 1 *
0011 1 *   THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0012 1 *   ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0013 1 *   INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0014 1 *   COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0015 1 *   OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0016 1 *   TRANSFERRED.
0017 1 *
0018 1 *   THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0019 1 *   AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0020 1 *   CORPORATION.
0021 1 *
0022 1 *   DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0023 1 *   SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0024 1 *
0025 1 *
0026 1 *****
0027 1
0028 1 ++
0029 1 FACILITY: VAX/VMS System Service Call Monitor
0030 1
0031 1 ABSTRACT:
0032 1
0033 1     This module makes a copy of System Service vector in P0 space,
0034 1     then modifies the System Service vector JSB into intercept code.
0035 1
0036 1     SSI.B32 is split into 2 portions: SSIK.B32 is strictly running
0037 1     in kernel mode to do the setup, SSIU.B32 is running in user
0038 1     mode only.
0039 1
0040 1 ENVIRONMENT:
0041 1
0042 1     VAX/VMS operating system, CMKRNL privilege required.
0043 1
0044 1 AUTHOR: David Thiel, 30-Dec-1981
0045 1
0046 1 Modified by:
0047 1
0048 1     Ping Sager, 19-Sep-1983
0049 1
0050 1 --
0051 1
0052 1
0053 1 Include files
0054 1
0055 1 LIBRARY 'SYSS$LIBRARY:LIB.L32';           ! VAX/VMS common definitions
0056 1 REQUIRE 'SRC$:SSIDEF.REQ';               ! Definitions for SSI
0146 1
```



```
59 0147 1 !
60 0148 1 ! Table of contents
61 0149 1
62 0150 1 FORWARD ROUTINE
63 0151 1     ssik_start,      ! Main routine
64 0152 1     make_p0_space, ! Allocate save/data/code area in P0
65 0153 1     ssik_setup;   ! Establish intercept
66 0154 1
67 0155 1 GLOBAL
68 0156 1     ssv_munged_flag: INITIAL(0), ! Set to TRUE when we actually
69 0157 1                                     change the system service vector.
70 0158 1     base : REF BBLOCK,           ! Address of system service vector
71 0159 1     intercept : REF VECTOR [, BYTE], ! Address of saved system vector,
72 0160 1                                     data, code area in P0
73 0161 1     range : VECTOR [2, LONG];    ! Pages allocated in P0
74 0162 1                                     (maps in ISSH.MAR)
75 0163 1
76 0164 1 ! This portion (SSIK.B32) of the SSI.B32 is strictly running in kernel mode,
77 0165 1 ! and we are not intercepting anything in kernel mode. So there is no
78 0166 1 ! need to have this flag to indicate this program is running. This flag
79 0167 1 ! is set to indicate the other part (SSIU.B32) is running which runs in
80 0168 1 ! user mode.
81 0169 1
82 0170 1 EXTERNAL      ! Flag set to indicate (SSIU.B32) is
83 0171 1     ssi_running_flag; ! running
84 0172 1
85 0173 1 OWN
86 0174 1
87 0175 1     l_base,           ! Variables (good for testing usage)
88 0176 1     l_intercept,      ! (resident) copy of base
89 0177 1     l_tvl,           ! (resident) copy of intercept
90 0178 1     range1 : VECTOR [2, LONG], ! (resident) copy of pointer
91 0179 1     range2 : VECTOR [2, LONG]; ! Maps in data area in range
92 0180 1                                     Created virtual space over system
93 0181 1                                     service vector
94 0182 1
95 0183 1 ! External routines
96 0184 1
97 0185 1 EXTERNAL ROUTINE
98 0186 1     SSI_USSK : ADDRESSING_MODE (GENERAL),
99 0187 1     sys$cretva : ADDRESSING_MODE (ABSOLUTE), ! Create virtual address space
100 0188 1     sys$qiov : ADDRESSING_MODE (ABSOLUTE), ! Base of transfer vector
101 0189 1     sys$rundwn : ADDRESSING_MODE (ABSOLUTE), ! Rundown
102 0190 1     issh_entry, ! Intercept code entry (ISSH)
103 0191 1     reset_ssv; ! Clean up the mess
104 0192 1
105 0193 1 EXTERNAL LITERAL
106 0194 1     issh_vec_length; ! Length of monitor code (ISSH)
107 0195 1
108 0196 1 EXTERNAL
109 0197 1     ctl$gl_ctlbasva : ADDRESSING_MODE (ABSOLUTE), ! Not used, if set, P0
110 0198 1                                     won't go away after image rundown
111 0199 1     issh_data_beg, ! Begin data area (template)
112 0200 1     issh_data_end, ! End data area (template)
113 0201 1     issh_prio_mask, ! Mask to control the calling of the
114 0202 1                                     user routines
115 0203 1     issh_vec_base, ! ISSH base address
```


SSIK
V04-000

E 3
15-Sep-1984 23:41:10 VAX-11 Bliss-32 V4.0-742 Page 3
14-Sep-1984 12:18:30 DISK\$VMSMASTER:[DEBUG.SRC]SSIK.B32;1 (2)

:	116	0204	1	issh_running_flag,
:	117	0205	1	
:	118	0206	1	issh_stack,
:	119	0207	1	
:	120	0208	1	
:	121	0209	1	issh_stkptr,
:	122	0210	1	ssi_table : VECTOR [, LONG];
:	123	0211	1	

:	Contain pointer to ssi_running_flag
:	in P0
:	Local stack in P0 to keep track
:	which user routine is active
:	at the moment
:	Pointer to the above local stack
:	Configuration SS data table


```
125 0212 1 GLOBAL ROUTINE ssik_start (RUNDWN_ADDR) =
126 0213 1
127 0214 1 ---
128 0215 1
129 0216 1 Function:
130 0217 1
131 0218 1 This is the main routine of the VAX/VMS System Service
132 0219 1 Monitor. It calls appropriate actions.
133 0220 1
134 0221 1 Inputs:
135 0222 1
136 0223 1 RUNDWN_ADDR : This address only can be rundown system index.
137 0224 1
138 0225 1 Outputs:
139 0226 1
140 0227 1 Worst status encountered.
141 0228 1
142 0229 1 ---
143 0230 1
144 0231 1 BEGIN
145 0232 1
146 0233 1 BUILTIN FP;
147 0234 1
148 0235 1 LOCAL
149 0236 1 prio_mask: REF VECTOR[,BYTE], ! Current mask value after set
150 0237 1 status; ! Return status
151 0238 1
152 0239 1
153 0240 1 ! Check for rundown case. The only way for this case to show up:
154 0241 1 SYSS$RUNDWN is called when image exits and intercept system service
155 0242 1 is setup. We simply put SSV back, and delete P0 space.
156 0243 1
157 0244 1 ! Note: next line is temporary, for I use the last 4 longwords in
158 0245 1 SSV itself to store some values.
159 0246 1 intercept = .(SYSS$QIOW + sgn$c_sysvecpgs * 512 - 4);
160 0247 1 IF .rundwn_addr EQL SYSS$RUNDWN
161 0248 1 THEN
162 0249 1 BEGIN
163 0250 1 IF .intercept EQL 0 THEN RETURN 0; ! Can't be possible.
164 0251 1 status = reset_ssv(); ! Call routine to Clean up.
165 0252 1 RETURN .status; ! Return Status. (Actually
166 0253 1 END; ! status is always 1).
167 0254 1
168 0255 1
169 0256 1 ! If this code is first time called, sets up the intercept, else simply
170 0257 1 returns.
171 0258 1
172 0259 1 IF .intercept NEQ 0 THEN RETURN ss$_normal;
173 0260 1
174 0261 1
175 0262 1 ! P0 space has not been set up by anyone yet. Grap some space.
176 0263 1 Set up SSV.
177 0264 1
178 0265 1 base = sys$qiow;
179 0266 1 status = make_p0_space();
180 0267 1 IF .status THEN status = ssik_setup();
181 0268 1 IF NOT .status THEN RETURN .status;
```



```
182 0269 2
183 0270 2
184 0271 2
185 0272 2
186 0273 2
187 0274 2
188 0275 2
189 0276 2
190 0277 2
191 0278 2
192 0279 2
193 0280 2
194 0281 2
195 0282 2
196 0283 2
197 0284 1
```

```
! Now that we have modified the system service vector, set the global flag
! which indicates that the system service vector has been modified. This
! flag gets cleared in RESETSSI.
```

```
ssv_munged_flag = 1;
```

```
! Initialize current mask to 0. (Assume nothing is active at
! this moment.
```

```
prio_mask = .intercept + issh_prio_mask - issh_vec_base;
```

```
.prio_mask = 0;
RETURN ss$normal;
END;
```

```
.TITLE SSIK
.IDENT \V04-000\
```

```
.PSECT $OWN$,NOEXE,2
```

```
00000 L_BASE: .BLKB 4
00004 L_INTERCEPT:
          .BLKB 4
00008 L_TVL: .BLKB 4
0000C RANGE1: .BLKB 8
00014 RANGE2: .BLKB 8
```

```
.PSECT $GLOBAL$,NOEXE,2
```

```
00000000 00000 SSV_MUNGED_FLAG::
          .LONG 0
00004 BASE:: .BLKB 4
00008 INTERCEPT::
          .BLKB 4
0000C RANGE:: .BLKB 8
```

```
.EXTRN SSI_RUNNING_FLAG
.EXTRN SSI_USSK, SYS$CRETVA
.EXTRN SYS$QIOW, SYS$RUNDWN
.EXTRN ISSH_ENTRY, RESET_SSV
.EXTRN ISSH_VEC_LENGTH
.EXTRN CTLSGL_CTLBASVA
.EXTRN ISSH_DATA_BEG, ISSH_DATA_END
.EXTRN ISSH_Prio_MASK, ISSH_VEC_BASE
.EXTRN ISSH_RUNNING_FLAG
.EXTRN ISSH_STACK, ISSH_STKPTR
.EXTRN SSI_TABLE
```

```
.PSECT $CODE$,NOWRT,2
```

```
00000000G 52 0000' CF 9E 00002
62 00000000G 9F D0 00007
8F 04 AC D1 0000E
```

```
.ENTRY SSIK_START, Save R2
MOVAB INTERCEPT, R2
MOVL @#SYS$QIOW+2556, INTERCEPT
CMPL RUNDWN_ADDR, #SYS$RUNDWN
```

```
: 0212
: 0246
: 0247
```

SSIK
V04-000

H 3
15-Sep-1984 23:41:10
14-Sep-1984 12:18:30

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[DEBUG.SRC]SSIK.B32;1

Page 6
(3)

		0A	12	00016	BNEQ	1\$		
		62	D5	00018	TSTL	INTERCEPT		0250
		3C	13	0001A	BEQL	3\$		
0000G	CF	00	FB	0001C	CALLS	#0, RESET_SSV		0251
			04	00021	RET			0252
		62	D5	00022	TSTL	INTERCEPT		0259
		2E	12	00024	BNEQ	2\$		
FC	A2	00000000G	8F	D0	00026	MOVL	#SYSSQIOW, BASE	0265
0000V	CF		00	FB	0002E	CALLS	#0, MAKE_PO_SPACE	0266
	24		50	E9	00033	BLBC	STATUS, 4\$	0267
0000V	CF		00	FB	00036	CALLS	#0, SSİK_SETUP	
	1C		50	E9	0003B	BLBC	STATUS, 4\$	0268
F8	A2		01	D0	0003E	MOVL	#1, SSV MUNGED_FLAG	0275
	50	0000G	CF	9E	00042	MOVAB	ISSH_Prio_MASK, R0	0281
	50		62	C0	00047	ADDL2	INTERCEPT, R0	
	51	0000G	CF	9E	0004A	MOVAB	ISSH_VEC_BASE, R1	
	50		51	C2	0004F	SUBL2	R1, Prio_MASK	
			60	D4	00052	CLRL	(Prio_MASK)	0282
	50		01	D0	00054	MOVL	#1, R0	0283
			04	00057	RET			
		50	D4	00058	CLRL	R0		0284
			04	0005A	RET			

; Routine Size: 91 bytes, Routine Base: \$CODE\$ + 0000


```
199 0285 1 ROUTINE make_p0_space =
200 0286 1
201 0287 1 ---
202 0288 1
203 0289 1 Function:
204 0290 1
205 0291 1     Create a save area for intercepting system services in P0
206 0292 1     space.
207 0293 1
208 0294 1 Inputs:
209 0295 1
210 0296 1     None.
211 0297 1
212 0298 1 Outputs:
213 0299 1
214 0300 1     status is returned.
215 0301 1
216 0302 1 ---
217 0303 1
218 0304 1 BEGIN
219 0305 1
220 0306 1 BIND
221 0307 1     exp_size = (issh_vec_length+XX'1FF') ^ -9;
222 0308 1                     ! ISSH.MAR code side
223 0309 1
224 0310 1 LOCAL
225 0311 1     status;
226 0312 1                     ! Return status
227 0313 1
228 0314 1     ! Create a save area to save the system vector, data area, and code
229 0315 1     ! in P0 space.
230 0316 1
231 0317 1     status = $EXPREG (
232 0318 1         PAGCNT = exp_size,
233 0319 1         REGION = 0,
234 0320 1         ACMODE = psl$C_kernel,
235 0321 1         RETADR = range);
236 0322 1     IF NOT .status THEN RETURN .status;
237 0323 1
238 0324 1     ! Map in from ISSH.MAR.
239 0325 1
240 0326 1     CH$MOVE (issh_vec_length, issh_vec_base, .range[0]);
241 0327 1
242 0328 1
243 0329 1     ! Set protection to saved area.
244 0330 1
245 0331 1     status = $SETPRT (
246 0332 1         INADR = range,
247 0333 1         PROT = prt$C_urkw);
248 0334 1     IF NOT .status THEN RETURN .status;
249 0335 1
250 0336 1
251 0337 1     ! Mapped in data area and control area.
252 0338 1
253 0339 1
254 0340 1     range1 [0] = .range [0] + issh_data_beg - issh_vec_base;
255 0341 1     range1 [1] = .range [0] + issh_data_end - issh_vec_base - 1;
```

```
.EXTRN  SYS$EXPREG, SYS$SETPRT
```

03FC 00000 MAKE_PO_SPACE:				WORD	Save R2,R3,R4,R5,R6,R7,R8,R9	
59	0000G	CF	9E 00002	MOVAB	ISSH_VEC_BASE, R9	0285
58	00000000G	00	9E 00007	MOVAB	SYSS\$ETPRT, R8	
57	0000'	CF	9E 0000E	MOVAB	RANGE, R7	
		7E	7C 00013	CLRQ	-(SP)	0321
		57	DD 00015	PUSHL	R7	
	000000000*	8F	DD 00017	PUSHL	#<<ISSH_VEC_LENGTH+511>a-9>	
00000000G	00	04	FB 0001D	CALLS	#4, SYSS\$EXPREG	
	56	50	D0 00024	MOVL	R0, STATUS	
	4E	56	E9 00027	BLBC	STATUS, 1\$	0322
00 B7	69	8F	28 0002A	MOV3	#ISSH_VEC_LENGTH, ISSH_VEC_BASE, @RANGE	0327
	7E	0E	7D 00031	MOVQ	#14, -(SP)	0334
		7E	7C 00034	CLRQ	-(SP)	
		57	DD 00036	PUSHL	R7	
	68	05	FB 00038	CALLS	#5, SYSS\$ETPRT	
	56	50	D0 0003B	MOVL	R0, STATUS	
	37	56	E9 0003E	BLBC	STATUS, 1\$	0335
	50	CF	9E 00041	MOVAB	ISSH_DATA_BEG, R0	0340
	50	67	C0 00046	ADDL2	RANGE, R0	
	51	69	9E 00049	MOVAB	ISSH_VEC_BASE, R1	
0000' CF	50	51	C3 0004C	SUBL3	R1, R0, RANGE1	
	50	CF	9E 00052	MOVAB	ISSH_DATA_END, R0	0341
	50	67	C0 00057	ADDL2	RANGE, R0	
	51	69	9E 0005A	MOVAB	ISSH_VEC_BASE, R1	
	50	51	C2 0005D	SUBL2	R1, R0	
	0000'	CF	FF A0 9E 00060	MOVAB	-1(R0), RANGE1+4	
		7E	04 7D 00066	MOVQ	#4, -(SP)	0348
		7E	7C 00069	CLRQ	-(SP)	
	0000'	CF	9F 0006B	PUSHAB	RANGE1	
	68	05	FB 0006F	CALLS	#5, SYSS\$ETPRT	
	56	50	D0 00072	MOVL	R0, STATUS	
	04	56	E8 00075	BLBS	STATUS, 2\$	0349
	50	56	D0 00078 1\$:	MOVL	STATUS, R0	
		04	0007B	RET		
FC A7	67	D0	0007C 2\$:	MOVL	RANGE, INTERCEPT	0354

SSIK
V04-000

K 3
15-Sep-1984 23:41:10
14-Sep-1984 12:18:30

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[DEBUG.SRC]SSIK.B32;1 Page (4) 9

50

01 D0 00080
04 00083

MOVL #1, R0
RET

: 0355
: 0357

; Routine Size: 132 bytes, Routine Base: \$CODE\$ + 005B

; 272 0358 1

```
274 0359 1 ROUTINE ssik_setup : PSECT (lkcode_1) =
275 0360 1
276 0361 1 ---
277 0362 1
278 0363 1 Function:
279 0364 1
280 0365 1     Setup System Service Intercept. Vector to be intercepted is
281 0366 1     in base, save/data area is in intercept.
282 0367 1
283 0368 1 Inputs:
284 0369 1
285 0370 1     Entry point address of this routine.
286 0371 1
287 0372 1 Outputs:
288 0373 1
289 0374 1     status is returned.
290 0375 1
291 0376 1 ---
292 0377 1
293 0378 1 BEGIN
294 0379 1
295 0380 1 LOCAL
296 0381 1     old_stat,           ! Old AST enable status
297 0382 1     status,            ! Return status
298 0383 1     temp_vec: VECTOR[2]; ! Parameter for $SETPRT
299 0384 1
300 0385 1 BIND
301 0386 1     tvl = base [sgn$c_sysvecpgs * 512, 0, 0, 0] : BBLOCK FIELD (tvb);
302 0387 1
303 0388 1
304 0389 1     l_base = .base;           ! SSV base address
305 0390 1     l_intercept = .intercept; ! Copied SSV base address
306 0391 1     l_tvl = tvl;             ! End of SSV
307 0392 1
308 0393 1
309 0394 1     ! Save original system vector in saved area. Disable AST first.
310 0395 1
311 0396 1     return if error (old_stat = $SETAST (ENBFLG = 0));
312 0397 1     CH$MOVE (sgn$c_sysvecpgs*512, .l_base, .l_intercept);
313 0398 1
314 0399 1
315 0400 1     ! Create virtual memory over the original system vector and copy
316 0401 1     ! the original contents back into it.
317 0402 1
318 0403 1     range2 [0] = .l_base;
319 0404 1     range2 [1] = .l_base + sgn$c_sysvecpgs*512 - 1;
320 0405 1     status = (sys$creta + (2X'80000000' - sys$qiow)) (
321 0406 1         range2,           ! inadr
322 0407 1         range2,           ! retadr
323 0408 1         0);              ! acmode?
324 0409 1
325 0410 1 IF NOT .status
326 0411 1 THEN
327 0412 1     BEGIN
328 0413 1
329 0414 1
330 0415 1     ! Enable AST.
```



```

      !
      ! IF .old_stat EQL ss$_wasset
      THEN
        return_if_error ($SETAST (ENBFLG = 1));

      RETURN .status;
      END;

      ! restore original contents of save/data area
      !
      CH$MOVE (sgn$_sysvecpgs*512, .l_intercept, .l_base);

      ! Set protection.
      !
      status = $SETPRT (
        INADR = range2,          ! pages to protect
        PROT = prt$_urkw);      ! kernel writable, others can read

      IF NOT .status
      THEN
        BEGIN

          ! Enable AST.
          !
          IF .old_stat EQL ss$_wasset
          THEN
            return_if_error ($SETAST (ENBFLG = 1));

          RETURN .status;
          END;

          ! Initialize local stack in P0.
          ! Stack pointer is 1 (points to the 1st element on stack),
          ! stack value is 0 (nothing is active at the moment).
          !
          .l_intercept + issh_stkptr - issh_vec_base = 1;
          .l_intercept + issh_stack - issh_vec_base = 0;

          ! Make the running flag user-writable.
          !
          temp_vec[0] = ssi_running_flag;
          temp_vec[1] = ssi_running_flag;
          return_if_error ($SETPRT (INADR = temp_vec, PROT = prt$_uw));

          ! Set up the running flag. If this flag is set, means that SSIU.B32
          ! is running so won't intercept any system service from that program.
          ! Otherwise, go ahead to intercept. For SSIU.B32 is running in user
          ! mode, which is the only mode we intercept.
          !
          .l_intercept + issh_running_flag - issh_vec_base = ssi_running_flag;

```

```
388 0473 2
389 0474 2
390 0475 2
391 0476 2
392 0477 2
393 0478 2
394 0479 2
395 0480 2
396 0481 2
397 0482 2
398 0483 2
399 0484 2
400 0485 2
401 0486 2
402 0487 2
403 0488 2
404 0489 2
405 0490 2
406 0491 2
407 0492 2
408 0493 2
409 0494 2
410 0495 2
411 0496 2
412 0497 2
413 0498 2
414 0499 2
415 0500 2
416 0501 2
417 0502 2
418 0503 2
419 0504 2
420 0505 2
421 0506 2
422 0507 2
423 0508 2
424 0509 2
425 0510 2
426 0511 2
427 0512 2
428 0513 2
429 0514 2
430 0515 2
431 0516 2
432 0517 1
```

```
! Set up 3 pointers at the end of the System Service Vector.
! - A pointer to the saved system service vector in P0.
! - A pointer to the saved intercept code entry point.
! - Address of the user defined system service.
tvL [ptr] = .L_intercept;
tvL [pg0] = .L_intercept + issh_entry - issh_vec_base;
tvL [pg1] = SSI_USSK;
DECR i FROM (.ssi_table[-1])/2 -1 TO 0 DO
  BEGIN
  BIND
    p = ssi_table [.i*2] : VECTOR [, LONG],
    t = .L_base + .p [1] : BBLOCK;

    ! Verify it is in System space.
    IF .p [1] GEQU %X'800'
    THEN
      0
    ELSE

      ! CALLS/CALLG intercepted at entry
      BEGIN
        t [2, 0, 8, 0] = op$.jsb; ! JSB
        t [3, 0, 8, 0] = %X'DF'; ! @W^ addressing mode
        t [4, 0, 16, 0] = tvL [pg0] - t [6, 0, 8, 0];
      END;
    END; ! End of DECR.

! Enable AST.
IF .old_stat EQL ss$_wasset
THEN
  return_if_error ($SETAST (ENBFLG = 1));
RETURN ss$_normal;
END;
```

```
.EXTRN SYS$SETAST
```

```
.PSECT LKCODE_1,NOWRT, SHR,2
```

```
OFFC 00000 SSIK_SETUP:
```

```
5B 00000000G 00 9E 00002
5A 00000000G 00 9F 00009
59 00000000 CF 9E 00010
```

```
.WORD Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11
MOVAB SYS$SETPRT, R11
MOVAB SYS$SETAST, R10
MOVAB L_INTERCEPT, R9
```

```
: 0359
```


56	0000*	5E	00000A00	08	C2	00015	SUBL2	#8, SP	:	
	FC	CF	0000*	8F	C1	00018	ADDL3	#2560, BASE, R6	:	0386
	04	A9		CF	7D	00022	MOVQ	BASE, L_BASE	:	0389
				56	D0	00028	MOVL	R6, L_TVL	:	0391
		6A		7E	D4	0002C	CLRL	-(SP)	:	0396
		58		01	FB	0002E	CALLS	#1, SYSS\$SETAST	:	
		5B		50	D0	00031	MOVL	R0, OLD_STAT	:	
		57	FC	50	E9	00034	BLBC	STATUS, -2\$	:	
00	B9	67	0A00	A9	D0	00037	MOVL	L_BASE, R7	:	0397
	10	A9		8F	28	0003B	MOVQ3	#2560, (R7), @L_INTERCEPT	:	
	14	A9	09FF	57	D0	00042	MOVL	R7, RANGE2	:	0403
				C7	9E	00046	MOVAB	2559(R7), RANGE2+4	:	0404
			10	7E	D4	0004C	CLRL	-(SP)	:	0405
			10	A9	9F	0004E	PUSHAB	RANGE2	:	
	00000000*	9F		A9	9F	00051	PUSHAB	RANGE2	:	
		57		03	FB	00054	CALLS	#3, @N<SYSS\$CRETVA+<-2147483648-SYSS\$QIOW>>	:	
		0E		50	D0	0005B	MOVL	R0, STATUS	:	
		09		57	E8	0005E	BLBS	STATUS, 1\$	:	0410
				58	D1	00061	CPL	OLD_STAT, #9	:	0417
				2F	12	00064	BNEQ	3\$	:	
		6A		01	DD	00066	PUSHL	#1	:	0419
		27		01	FB	00068	CALLS	#1, SYSS\$SETAST	:	
				50	E8	0006B	BLBS	STATUS, 3\$	:	
					04	0006E	RET		:	0421
FC	B9	00	B9	8F	28	0006F	MOVQ3	#2560, @L_INTERCEPT, @L_BASE	:	0427
			7E	0E	7D	00077	MOVQ	#14, -(SP)	:	0434
				7E	7C	0007A	CLRQ	-(SP)	:	
				A9	9F	0007C	PUSHAB	RANGE2	:	
		6B		05	FB	0007F	CALLS	#5, SYSS\$SETPRT	:	
		57		50	D0	00082	MOVL	R0, STATUS	:	
		11		57	E8	00085	BLBS	STATUS, 4\$	:	0436
		09		58	D1	00088	CPL	OLD_STAT, #9	:	0443
				08	12	0008B	BNEQ	3\$	:	
				01	DD	0008D	PUSHL	#1	:	0445
		6A		01	FB	0008F	CALLS	#1, SYSS\$SETAST	:	
		3F		50	E9	00092	BLBC	STATUS, 5\$	:	
		50		57	D0	00095	MOVL	STATUS, R0	:	0447
					04	00098	RET		:	
		50	0000G	CF	9E	00099	MOVAB	ISSH_STKPTR, R0	:	0455
		50		69	C0	0009E	ADDL2	L_INTERCEPT, R0	:	
		51	0000G	CF	9E	000A1	MOVAB	ISSH_VEC_BASE, R1	:	
		50		51	C2	000A6	SUBL2	R1, R0	:	
		60		01	D0	000A9	MOVL	#1, (R0)	:	
		50	0000G	CF	9E	000AC	MOVAB	ISSH_STACK, R0	:	0456
		50		69	C0	000B1	ADDL2	L_INTERCEPT, R0	:	
		51	0000G	CF	9E	000B4	MOVAB	ISSH_VEC_BASE, R1	:	
		50		51	C2	000B9	SUBL2	R1, R0	:	
				60	D4	000BC	CLRL	(R0)	:	
		6E	0000G	CF	9E	000BE	MOVAB	SSI_RUNNING_FLAG, TEMP_VEC	:	0461
	04	AE	0000G	CF	9E	000C3	MOVAB	SSI_RUNNING_FLAG, TEMP_VEC+4	:	0462
		7E		04	7D	000C9	MOVQ	#4, -(SP)	:	0463
				7E	7C	000CC	CLRQ	-(SP)	:	
			10	AE	9F	000CE	PUSHAB	TEMP_VEC	:	
		6B		05	FB	000D1	CALLS	#5, SYSS\$SETPRT	:	
		7C		50	E9	000D4	BLBC	STATUS, 9\$	:	
		50		69	D0	000D7	MOVL	L_INTERCEPT, R0	:	0471
		51	0000GCF	40	9E	000DA	MOVAB	ISSH_RUNNING_FLAG[R0], R1	:	

		52	0000G	CF	9E	000E0	MOVAB	ISSH_VEC_BASE, R2	
		51		52	C2	000E5	SUBL2	R2, R1	
		61	0000G	CF	9E	000E8	MOVAB	SSI_RUNNING_FLAG, (R1)	
	FC	A6		50	D0	000ED	MOVL	R0, -4(R6)	0479
		51	0000G	CF	9E	000F1	MOVAB	ISSH_ENTRY[R0], R1	0480
		50	0000G	CF	9E	000F7	MOVAB	ISSH_VEC_BASE, R0	
F8	A6	51		50	C3	000FC	SUBL3	R0, R1, -8(R6)	
		A6	00000000G	00	9E	00101	MOVAB	SSI_US\$K, -12(R6)	0481
		51		02	C7	00109	DIVL3	#2, SSI_TABLE-4, R1	0483
	F4	50		51	D0	0010F	MOVL	R1, 1	0487
	0000G			2C	11	00112	BRB	7\$	
		51		01	78	00114	ASHL	#1, 1, R1	0486
		52	0000G	CF	41	DE	MOVAL	SSI_TABLE[R1], R2	
		A9	04	A2	C1	0011E	ADDL3	4(R2), L_BASE, R1	0487
	00000800	8F	04	A2	D1	00124	CMPL	4(R2), #2048	0492
				12	1E	0012C	BGEQU	7\$	
	02	A1	DF16	8F	B0	0012E	MOVW	#57110, 2(R1)	0502
		52	F8	A6	9E	00134	MOVAB	-8(R6), R2	0504
		52		51	C2	00138	SUBL2	R1, R2	
04	A1	52		06	A3	0013B	SUBW3	#6, R2, 4(R1)	
		D1		50	F4	00140	SOBGEQ	1, 6\$	0483
		09		58	D1	00143	CMPL	OLD_STAT, #9	0511
				08	12	00146	BNEQ	8\$	
				01	DD	00148	PUSHL	#1	0513
		6A		01	FB	0014A	CALLS	#1, SYSS\$SETAST	
		03		50	E9	0014D	BLBC	STATUS, 9\$	
		50		01	D0	00150	MOVL	#1, R0	0515
				04	00153	9\$:	RET		0517

; Routine Size: 340 bytes, Routine Base: LKCODE_1 + 0000

:	433	0518	1
:	434	0519	1 END
:	435	0520	0 ELUDOM

PSECT SUMMARY

Name	Bytes	Attributes
\$GLOBALS	20 NOVEC, WRT, RD	, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$OWNS	28 NOVEC, WRT, RD	, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODE\$	223 NOVEC, NOWRT, RD	, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
LKCODE_1	340 NOVEC, NOWRT, RD	, EXE, SHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	----- Symbols -----		Pages Mapped	Processing Time
	Total	Loaded Percent		

SSIK
V04-000

^{0 4}
15-Sep-1984 23:41:10
14-Sep-1984 12:18:30

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[DEBUG.SRC]SSIK.B32;1 Page 15
(5)

:
: _\$255\$DUA28:[SYSLIB]LIB.L32;1 18619 12 0 1000 00:01.8

:
: COMMAND QUALIFIERS

: BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:SSIK/OBJ=OBJ\$:SSIK MSRC\$:SSIK/UPDATE=(ENH\$:SSIK)

: Size: 563 code + 48 data bytes
: Run Time: 00:12.6
: Elapsed Time: 00:40.1
: Lines/CPU Min: 2474
: Lexemes/CPU-Min: 15925
: Memory Used: 131 pages
: Compilation Complete

0101 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

